Protocol Analysis of Ubiquitous Sensor Networks with Health Care Monitoring and Surveillance

David Dahlin ic08dd7@student.lth.se
Pétur G. Hjartarson ic08ph0@student.lth.se

Department of Electrical and Information Technology
Lund University

Advisor: Maria Kihl, EIT.

July 25, 2013

Printed in Sweden
E-huset, Lund, 2013

Abstract

As the population grows and people live longer, the demand of health care will increase. Visits to the hospital are exhausting and time consuming, especially for patients with long term diseases. The long term disease patients and elderly take up the largest part of the hospital resources and when the knowledge of sensor networks is improving and the communication channels are getting more reliable, the need for visiting the hospital in order to do a simple measurement can be reduced. Many measurements and even some monitoring activities can therefore be done at home instead of taking time and resources from the hospitals. Not only can this prove to be beneficial in terms of cost, but also when it comes to produce new services and business models.

This thesis is about implementing such a feature in an existing sensor network that is being used mainly for home security. It will present a test implementation, or a proof of concept that it is possible to send arbitrary data at the time being and it will also compare the existing sensor network protocol with other sub gigahertz sensor network protocols. Furthermore the thesis will address a system model based on the standard ISO/IEEE 11073 PHD to make it possible to co-exist with manufacturers of different health devices.

Keywords: Ubiquitous health care, Health care sensor networks, ISO/IEEE 11073 PHD, ZigBee® HC

Sammanfattning

När populationen växer och livslängden på befolkningen ökar, blir även kraven på vården större. Många besök till vården är utmattande, speciellt för patienter som är långtidssjuka. Det är de långtidssjuka och äldre som tar upp största delen av vårdens resurser och när kunskapen om sensornätverk blir bättre och kommunikationskanalerna blir säkrare, så minskar behovet att åka till sjukhuset för att göra enklare mätningar. Många mätningar och även viss övervakning kan istället göras direkt i hemmet vilket sparar tid och resurser hos sjukhusen.

Detta examensarbete handlar om att implementera en sådan funktion i ett redan existerande sensornätverk som har använts främst för säkerhet i hemmet. Uppsatsen kommer att presentera en testimplementation, eller snarare ett konceptbevis som visar att det är möjligt att skicka godtycklig data, där vi även provar att skicka autentisk medicindata från en hemoglobinmätare. Den kommer även att jämföra det redan existerande industri accepterade radio protokollet vid namn ZigBee med ett annat sub gigahertz radio protokoll som är utvecklat av Securitas Direct AB (Verisure). Uppsatsen kommer även att redogöra för en systemmodell baserat på standarden ISO/IEEE 11073 PHD för att göra det möjligt att samexistera med andra tillverkare av olika vårdutrustningar.

Implementationen visade sig fungera väl, men för att uppnå de krav som ställs på vårdutrustning måste vidareutveckling ske på systemet. En stor del av resultatet visade sig vara bristen på ett separat applikationslager, dvs. utan att implementera exakt vilka kommandon som kan skickas, kan man välja att starta en dataström med godtycklig data. Detta är något radioprotokollet ZigBee® kan hantera. ZigBee® har även en egen så kallad applikationsprofil, ZigBee®Health Care, som har stöd för medicinstandarden ISO/IEEE 11073 PHD. Applikationsprofilen är utformad för att göra det lättare för tillverkare av olika medicinutrustningar att kunna kommunicera med varandra.

Acknowledgements

We want to thank the following for guidance and help during the project.

- **Maria Kihl** for being a good supervisor and leading us towards our results.
- **Mark Spånberg** for helping us on behalf of the companies interests.
- **Filip Skarp** for showing interest in us and making it possible to start with the project.
- **Noroz Akhlagi** for coordinating us on behalf of the project.
- **Verisure** for letting us do research on behalf on their company.
- **Ragnhildur Kristjansdóttír** For her strength during the last six months and showing Pétur courage, ambition and hope.
- **Hjörtur Oddsson** For his great support to Pétur and showing out directions at academic and life crossroads.
- **Mats Dahlin** For great guidance and advice during David's academic run.
- **Rosa Maria Dahlin** For support, encouragement and being there when needed.
- **Isabel Dahlin** For the support and understanding.
- **Our friends** for putting up with us during the evenings and weekends.

David Dahlin and Pétur G. Hjartarson Lund, Midsommarafton 2013

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The population is growing, and as we are getting older it is expected that the load on the hospitals will increase. The Swedish Government Office predicts a couple of future scenarios which will require that people will need to take care of their own measurements in their own homes. The thesis will focus on getting medical equipment to communicate within a sensor network that has the main feature of being an alarm system.

1.1 Background

A research project named itACiH, IT support for Advanced Cancer care in the Home, recently started in the Skåne region. The project involves both the industry, hospitals as well as Lunds University. It is funded by Vinnova which sponsored the project with 10 million SEK to speed up the phase of new health care techniques that work in a palliative manner at home. The aim of the project is to reduce the number of patients who need to transport themselves to clinics, health centers and hospitals by providing medical equipment. By doing this the number of visits will be reduced since many of these measurements can be done at home or more locally if the equipment is easy to use, has high reliability and is secure enough for transmitting and receiving potentially sensitive data.

1.2 itACiH and PalCom

The Vinnova project, itACiH focuses on cancer patients and partly relies upon a software called PalCom that was developed to create software to connect equipment or devices which are not designed to communicate with each other. It serves as a protocol bridge or translation program. The main focus of the research project is *heterogeneous networks*, i.e. that different computers with different operating

systems can communicate, *incompatible devices*, i.e. creating a bridge between devices that use different interfaces and protocols to be able to communicate and create *architecture support*.

The PalCom software is a middleware for Pervasive Computing, in other words, it is developed in a way that the need for standards is minimised. The only remark is that it requires a reliable transport layer. The PalCom architecture separate Computation (realized by service) from Assemblies that cover Configuration (what services are included) and Coordination (how the Services interact). Communication in PalCom is providing a unified view of heterogeneous networks and use discovery mechanism to make assemblies to react to a changing environment as services come and go[11].

This thesis has no focus on the PalCom software, instead it is focused on how the sensor network of the Securitas Direct AB (Verisure) should be able to transport the data from the different medical devices in a reliable way to the receiver on the other side of the back end. Also, to be compatible with other projects in other countries, not only itACiH, focus has been on what kind of sensor network that would be most suitable for such data transmission and also to analyse if there are existing standards that could support new business opportunities.

1.3 Medical Motivation

The number of cancer cases is expected to double over the the coming 20 years. In addition, the number of patients wanting to receive medical attention at home has increased[4]. This allows for an opportunity not only for the patient, but also when it comes to the administrative costs when giving care on site. Therefore reducing costs can decrease the load and benefit both staff and patient care. The task is to be able to gather patient data, monitor and to be able to communicate with the patient when needed. By allowing remote patient care the process of chemotherapy, sampling and follow up could be performed off site and hopefully in the long term, patients with other conditions can receive similar medical attention.

1.3.1 The need for Ubiquitous Health Care

In 2050, the average life time for people over 65 will rise with 3.2 years, from 83.1 to 86.3 for men and 2.2 years for women from 86 to 88.2 years. These numbers are based on a dynamic model called SESIM and is built on a simulation of 300 000 individuals in Sweden[4].

The report from The Swedish Government Office[4] predicts four scenarios that will occur to the Swedish population after 40 years:

- The population will grow.
- The population will get older.
- The ambition of health care will grow.
- The health of people will change.

This will result in higher load on the existing hospitals since the majority of the people with poor health is elderly. This is why automated ubiquitous health care will grow rapidly in the future[14]. By allowing people to perform simple measurements at home such as measuring hemoglobin levels, blood pressure and insulin levels the cost and time needed by medical personnel will be less. Moreover people who require continuous surveillance can be monitored through these automated systems. One can also control medication and nutrition that is being given intravenously.

Additionally the workload on nurses and physicians will decrease and staff with less experience from handling patients directly can be hired for monitoring purposes which will lower the cost per patient[19].

1.3.2 Health Care and Sensor Networks

The goal of ubiquitous health care sensor networks is to reduce the workload in public health care by minimizing the number of patients needing help with relatively simple measurements. This can be done with sensor networks, that can be controlled with mobile devices from remote clinics. At the same time, patients can be monitored in a way where detected aggravated health status is discovered early[13].

In order to implement such a sensor network, some problems have to be taken in consideration.

- Correction techniques for the data to be transmitted need to be thoroughly implemented. The data is sensitive and needs to be correct to have such a health care system.
- Power saving techniques to minimize the energy consumption for the nodes. The technique must apply when the electricity at home is out of order, as well as have long up-time on the products at home.
- Sophisticated data analysis is needed in order to handle a large number of data measurements. It is also important to have good analysis for following

health issues, for example, both detecting when a flu has its outbreak and also for tracking how different diseases behave in order to confirm a diagnose as early as possible.

1.4 The aim of the thesis

The aim of this thesis is to present a proof of concept to make a realization of forth coming topics and prove the feasibility. This product will probably not be released to the market, however, it will help future development.

1.5 Questions

Is there a possibility to use the existing protocol of Securitas Direct AB (Verisure) with a proprietary protocol to send and receive health data from and to the existing database to provide surveillance and control of medical equipment in the homes of patients?

1.5.1 Development questions

What is needed in order to be able to send the medical data?

- Is it enough to alter the existing commands to be interpreted as other commands in the back end of Securitas Direct AB (Verisure)?
- Is ASCII commands sufficient for the given purpose of the commands?
- What will be required from an application to send/receive data?

1.6 Relevant work

Through out the years, there have been a lot of improvement on the sensor networks and the treatment of data between different units. However, projects have not merged into one single project until now, and the focus has previously not been on interoperability. Instead different institutes have been developing their own way to realize their vision of Home care.

There have been different systems, for example the Code blue project, developed by Harvard University started in 2004 where the development ended in 2007 and where the system had no focus on interoperability. The Virginia University also had health care devices called in-home monitoring system, but both these systems are no longer being developed[13].

There have been European institutes that have been focusing on the same matter, such as the WohnSelbst project[2] based on the program MCPlus of the Horst Schmidt Clinic. It did not offer in home monitoring but it was a program focused on the health of elderly by offering; one health check per year, electronic patient record, access to health care competence center for advice and dedicated services in hospitals.

The WohnSelbst project should add additional services in form of regular checks of vitals (eg. weight, blood measurements, glucose levels), coaching patients in case of disease according to critical measurements and guidance of patients by a competence center. Today the WohnSelbst project has gone towards an implementation of the ISO/IEEE 11073 based Continua Health Alliance certification.

Chapter 2

Methodology

This chapter will describe what methods and tools that have been used in order to make the protocol analysis and test implementation. It will give an understanding of what parts we have chosen to work more with and what parts that have been excluded.

2.1 Theory

In order to reach a level of abstraction of this thesis, relevant research was carried out by gathering models, outlines and technical solutions with up to date wireless sensor networks. The work consisted of analysing a proprietary protocol and describe the relevance of the current protocol compared to existing industry accepted standards by theory and practice. This was the foundation for our test implementation and evaluation of our own developed proprietary based solution.

2.2 Meetings

We have on a continuous basis during the project had meetings with our advisors at Securitas Direct AB (Verisure) and with our advisor at the department. Also, we have had a meeting with Boris Magnusson who is the head of the research project itACiH.

2.3 Practice

To be able to understand how the system of Securitas Direct AB (Verisure) functions some practice and hands-on experience was needed. Practice was laid on the proprietary protocol of Securitas Direct AB (Verisure) and also the communication

between the Gateway and Securitas Direct AB (Verisure) back end. The documentation given by the development department of Securitas Direct AB (Verisure) was satisfactory. In order to understand the system completely we had to make a test implementation of a system that could deliver arbitrary data.

However, there were some restrictions on the system. The proprietary based sensor network does not have any arbitrary data fields or in other words the OSI application layer is not treated separately. The stack is built in a way that merges the application layer with all the layers to the transport layer. The test board that we have worked with is a Texas Instrument CC1110 Radio Frequency (RF) Systems-on-Chip (SoC) which includes the radio layer, the wireless layers and the application layer into one functioning chip with RF capabilities. The proprietary protocol is based on a set of commands to alter different states of the nodes. Since this is an alarm system, the requirement on the security is very high. Also, Securitas Direct AB (Verisure) has restrictions on what kind of nodes that can exist within the sensor networks. All the nodes of the sensor network have battery restrictions. However, our test implementation did not have such restrictions.

2.3.1 Transfer of data

The wireless sensor network (WSN) that we have worked with treat the data uploaded and downloaded in different manners. When wanting to send data to the node (download) and when receiving data from the node (upload) one must to initiate different commands and set the SoC in different states.

2.3.1.1 Upload

The upload part uses a command that is used for sending faults logs from the device. It is implemented in the protocol and utilizes four bytes of arbitrary data within the WSN. In our case, we chose to represent the data we want to send as decimal coded ASCII characters. The data from the Gateway to the back end of Securitas Direct AB (Verisure) is sent with an UDP packet.

2.3.1.2 Download

The download is treated in a more complex matter. The only way to send arbitrary data to the SoC is to let the system wrongly identify us as a keypad belonging to the home security system. By doing a Representational State Transfer, REST, to the back end, we can alter the existing keypad configuration command to be able to access node through the Gateway.

2.4 Tools

For the development and research we used the following development tools. Note that order does not display importance.

- **Eclipse** Classic 4.22 Helios: For developing the frontend where the REST calls and the database queries are being made.
- **IAR** Embedded Workbench 8.x for 8051. It is a workbench for developing embedded systems, in our case it was used to flash and implement the features of the CC1110 SoC.
- **RF Analyser** A PC software application that can display and store radio packets captured by a listening RF device.
- **Verisure Mypages web page** A tool for managing your home Verisure system.
- **Db visualizer** A database management and analysis tool for all major databases (e.g. Oracle, SQL Server, DB2, Sybase, MySQL, SQLite).
- **ShareLaTeX.com** For the making of the report.
- **Wireshark** A network protocol analyzer for Unix and Windows.

Chapter 3

Sensor Networks

This chapter describes and gives a general picture of what Wireless Sensors Networks (WSN) are, how they can be applied and which protocols allow for this type of network environment.

3.1 Introduction

The design of low power networks have been carried out to provide high wireless connectivity between products with limited battery life time spanning from month to years. To achieve battery times of this duration the use of the high energy consuming transceiver must transmit and receive with sparse schemes not only to obtain the required battery time, but also operate under certain frequency regulations. The distance a signal can be received and transmitted can be described as a function of the power input to corresponding antenna. This makes the wireless network range between individual nodes to be somewhat limited.

3.2 Components

A WSN is a network consisting of autonomous sensors (or nodes) each with the purpose to monitor physical or environmental conditions (see Figure 3.1). It can for example be temperature, sound, pressure, humidity or other relevant measurements. The collected data is then passed on to a main location (e.g gateway) that handles and controls the data. Data can be sent in both directions hence enabling condition based control activity to the sensors.

A WSN can be applied to a number of different environments and implemented according to different standards and platforms. One of many recent developments in the last couple of years has been the focus of developing smart homes with help from WSNs.

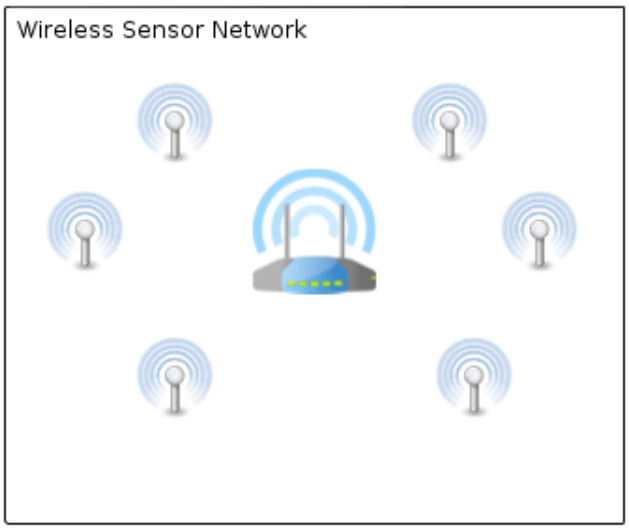

Figure 3.1: A Wireless Sensor Network with a Node and a Gateway

WSN can exist from scales from a few up to hundreds of nodes. There exist networks where the nodes are interconnected referred to as multi-hop nodes. Each sensor node is typically equipped with a transceiver with either external and internal antenna, a micro controller which is an electrical circuit and interface to the different sensors, and a power source which tends to be a battery or an energy harvesting unit.

3.2.1 Node

A typical sensor node can be considered a minimalistic computer consisting of a central processing unit with limits regarding its computational power and memory. The unit has also communication device in terms of a wireless transceiver. Sensor nodes are often battery powered which will require power efficient implementation and communication.

3.2.2 Gateway

The sensor node communicates with a base station, also referred to as the gateway which usually has more capacity when it comes to computational power and other requirements regarding power consumption. All data from the sensor node is passed on to the gateway and most commonly sent to a server.

3.3 Standards for Sensor Networks

There are several standards under development and a range of different standardization bodies are under way when it comes to the field of WSNs. For instance the professional association IEEE 802.15 is mainly focusing on the standardization of the physical and MAC layer. In addition to this there are a number of non-standard proprietary protocols being developed focusing on higher layers for special end-need purposes. ZigBee® is an example of this.

3.4 ZigBee®

ZigBee® is a high level communication suite specification that allows for communication over low power, small digital based radio in IEEE 802 standard for personal area networks (PAN). It is often applied to mesh networks for transmission of data over a long range, accomplished by sending data between intermediate connected devices.

By defining an already existing communication basis for network management in the 802.15.4 standard, flexible higher layer network implementation can be carried out, thus providing a base for higher level protocols such as ZigBee®.

3.4.1 Network Topology

There are currently three types of network topologies available in the ZigBee® specification.

The tree network topology is where each node in the network associates itself to a parent node and addressing is done in a similar fashion as an internet IP address. This opens up for several possibilities where increased efficiency is one of them when it comes to routing algorithms.

A star network is where a designated node acts as a central network coordinator which is responsible for different network management controls which includes node associations, network joining, setup of linking permissions, message handling and security parameters. If a sudden disruption causes the star node to go down no communication can be made which implies high reliability of the star node to be operable at any given time.

The last network topology can be formed as a mesh network which in a general form can be described as a network to which there is a minimum of two pathways to each node. In this sense every node is directly connected other every other node in the network[1].

3.4.2 ZigBee® stack architecture

The ZigBee® stack architecture (see Figure 3.2) consists of set layers with each one having its specific service and operation performed for the layer above. There are data entities that provide data transmission services and management entities that handle all other kind of operations. All the different layers are interconnected upwards through service access points (SAP) that provide a set of function primitives to fulfill the required task functionality.

The stack is based on the seven layer model OSI, but in reality does ZigBee® handle specific layers relevant for the intended application. The IEEE 802.15.4-2003 defines the two sublayers PHY and MAC on which the ZigBee® foundation is built upon by adding a network layer (NWK) and an application layer which defines an application support layer (APS), ZigBee® device objects (ZDO) and so called manufacturer specific application objects[1].

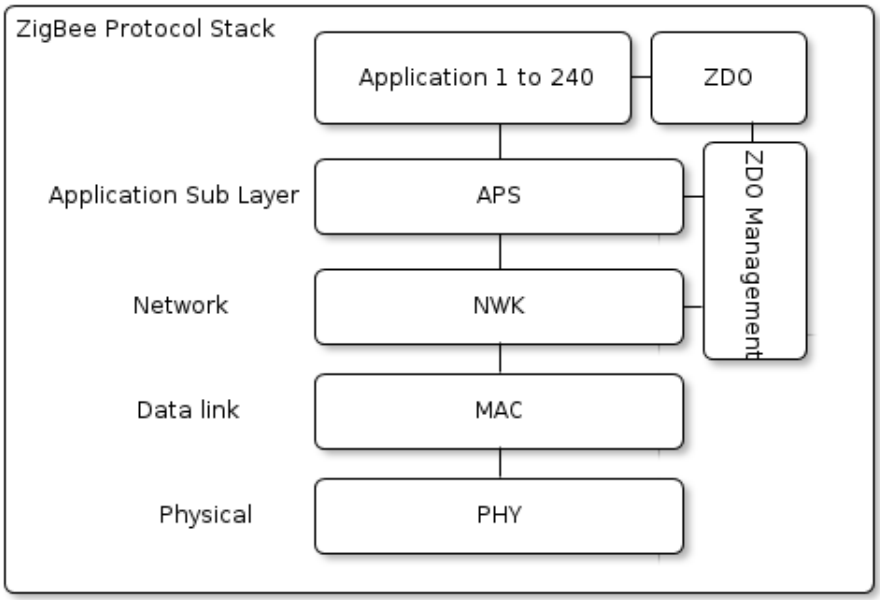

Figure 3.2: The ZigBee® protocol stack

3.4.2.1 PHY Layer

The physical layer, or more commonly known as the PHY Layer defines the physical links and on which frequencies and modulation techniques that are used for different countries and operations. The PHY layer provides the service of trans-

mitting data between different nodes in a network using a Direct-sequence spread spectrum (DSSS) modulation scheme and specifies communication data rates of 20 or 40 kbps for the 868/915MHz frequency channels and 250 kbps for the 2.4 GHz channels of operation. The frequency ranges are for European countries (868 MHz), U.S. (915 MHz) and worldwide (2.4 GHz) Industrial, Scientific, and Medical (ISM) frequency bands[1].

3.4.2.2 MAC Layer

Specified in IEEE 802.15.4-2003 RFC. Briefly, the MAC layer of the protocol includes features that involves reliable peer-to-peer communication. For example packet, frame management, node associations and acknowledgements.

3.4.2.3 NWK Layer

Since the ZigBee® stack architecture is based on the IEEE 802.15.4 standard which only defines the MAC and physical layer the ZigBee® alliance has defined the network layer and application framework that is based on the two layer communication suite.

The purpose of the network layer is to provide functions that make correct use of the MAC sub layer and through interfaces provide support for above layers. Namely, the network layer has two functions:

- Provide a network management entity which is interface through NLDE-SAP to application layer
- Provide a network data entity which uses NLDE-SAP interface to upper application layer.

3.4.2.4 NWK management service

The NWK layer management entity SAP (NLME-SAP) enables the transport of management commands between the next upper layer and the network layer management entity.

3.4.2.5 Network data service

The network data entity is responsible for transport of application protocol data units, abbreviated APDUs between peer application entities. In this entity a set of different primitives are defined, each with its designated purpose. The primitives are constructed and forwarded to the interface NLDE-SAP of the upper layer. When a ZigBee® device has established a successful connection to the network,

data transmission may be proceeded. The NWK layer executes this by sending a NLDE-DATA.request to send a transmission request. When it is the other way around (i.e confirming a data request) the NWK layer sends NLDE-DATA.confirm to return a transmission request[1].

3.4.2.6 Application Support Sub-layer

Application support sublayer (ASP) is a layer functioning as an interface between the application layer and the network layer. It provides a general set of services accessible for use by the ZigBee® device object (ZDO) and manufacturer specific application objects. The services can be accessed through two different entities, the data service and the management service, each one accessible through their corresponding SAP. The functionality provided by the data service entity are the following:

- Binding
- Group address filtering
- Reliable transport
- Duplicate rejection
- Fragmentation

The Application Support Sub-Layer Management Entity (APSME) provides a management service to allow an application to interact with the stack. Examples of these functions are security, group management and other management services[1].

3.4.2.7 ZigBee® Device Object

The ZigBee® device object framework is an application object residing in the APL (see Figure 3.2). This object is responsible for setting upp configuration parameters set by the application or the stack profile. Its purpose is to initialize the APS, NWK layer and SSP and additional ZigBee® device layers except for already existing end applications through endpoints 1-240. The ZDO basically represents a predefined base class of functionality in which all applications are written upon. This allows for an abstraction for developers to not bother about underlying layers. The ZDO public interface handles exchange between application objects, application profiles and the APS.

3.4.2.8 Application Layer

The application layer is the highest level defined by the ZigBee® specification. It is where one effectively communicates with the interfaces to the ZigBee® system.

It consists of the majority of components specified in the ZigBee® specification. The ZDO with its management services in addition with the application objects determined by the manufacturer are seen as a part of this layer[1].

3.5 SimpliciTI™ protocol

We use this section to describe briefly how the under laying layers works of Securitas Direct Safety Radio Protocol, since these layers of the protocol stack are guarded by confidentiality.

The SimpliciTI™ network protocol is an open proprietary low-power RF protocol used in small simple RF networks usually consisting of less than 100 nodes. The SimpliciTI™ network protocol was developed by Texas Instruments to allow rapid implementation with minimal microcontroller resource requirements.

SimpliciTI™ provides three different main functions:

- Low memory requirements
- Network control in terms of security and and agile selectivity of radio frequency
- Support of sleeping modes

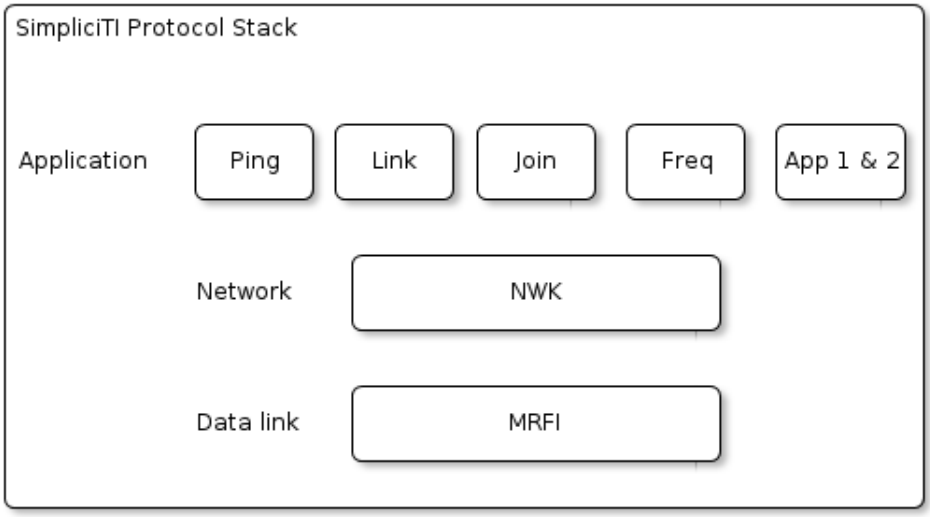

Figure 3.3: The SimpliciTI™ protocol stack

3.5.1 Protocol Architecture

The protocol (see Figure 3.3) is in comparison to the traditional seven layer OSI model a more downscaled approach with less layers. The application layer, which is the most significant, consists of a network application layer and a user application layer. The network application layer handles the data packet structure and is responsible for the process of the data message. The application layer basically processes directly the received or transmitted data.

SimpliciTI^{TM} consists of three layers: Data Link/Physical, network, and a application layer. The latter is the only layer that the developer needs to focus on for implementational needs. Based on this layer one develops the application (to be able to manage sensors), and implements network communication by using SimpliciTI^{TM} network APIs or network applications throughout.

The major differences between this protocol and the conventional OSI-model are the following[10]:

- In the SimpliciTI^{TM} communication model no formal PHY or Data Link (MAC/LLC) Layer exists. The data received directly from the radio is already framed so the radio performs these functions.
- The security support is implemented as a peer of the network layer due to the lack presentation layer where this function is implemented in the OSI model.
- It is in the application layer that the developer is required to implement reliable transport if needed due to the lack of an existing transport layer.

3.5.2 Topology

The protocol supports two basic topologies: a strictly peer-to-peer and a star topology in which the star hub is a peer to every other device. SimpliciTI^{TM} allows user to implement three different device types:

- End device
- Access Point
- Range extender

The end device is the base element of the network. It generally supports most of the sensors or actuators of the network. A strictly peer-to-peer network is exclusively composed of end devices (and eventually range extenders).

The access point supports features and functions such as store-and-forward support for sleeping End Devices, management of network devices in terms of

membership permissions, linking permissions, security keys, etc. The Access Point can also support End Device functionality. In the star topology the Access Point acts as the hub of the network.
Range extenders are intended to repeat frames in order to extend the network range. Due to their function, they are always on in order to fulfill this range extension task[9].

3.5.2.1 Minimal RF Interface

The Minimal RF Interface (MRFI) layer basically describes the read/write interface to the radio. Different radios supported by SimpliciTI^{TM} require different implementations but the basic interface offered to the network layer is the same for all radios. Different radios offer different levels of support for typical data link and physical layer responsibilities. MRFI encapsulates these differences.[9]

3.5.2.2 Network

The network (NWK) layer manages the received and transmitted queues and dispatches data frames to their corresponding destination. The destination is always an application designated by a port number. Network applications are internal peer-to-peer objects intended to manage network. They work on a predefined port and are not intended to be used by the developer (except ping for debugging purposes). Their usage depends on the SimpliciTI^{TM} device type[10]. The different network assigned port numbers are:

- **Ping (Port 0x01):** is used to detect the presence of a specific network device.
- **Link (Port 0x02):** used to support the link management of between two peers.
- **Join (Port 0x03):** used to guard entry to the network in topologies with APs.
- **Security (Port 0x04):** is used to alter security information such as encryption keys and encryption context.
- **Freq (Port 0x05):** used when to perform change channel, channel change request or a so called echo request.
- **Mgmt (Port 0x06):** used for general management of the device[10].

3.5.3 API commands

APIs is a way to allow the user to implement a reliable network with minor effort. But one must bear in mind have that the resulting network sacrifices flexibility for simplicity. Here are the different APIs supplied by SimpliciTI^{TM} :

- **initialization** : Used upon adding a device to a network which uses system parameters to setup a connection.
- **linking (bi-directional by default)**: Linking is used to associate devices with each other to allow forwarding of packets. Unlinking is the opposite process, e.g tearing down a link to a neighbour node.
- **peer-to-peer messaging**: Peer to peer messaging is possible by sending messages of variable length with the obtained LinkID.
- **configuration**: Local static peer configuration can be performed on a peer for example to configure an already existing peer connection. Other parameters can also be set during run time[10].

3.5.4 Securitas Direct Safety Radio Protocol

As described earlier, some layers are guarded by confidentiality and SimpliciTI^{TM} have been used in a informative matter to give a brief understanding about the under laying layers of the protocol stack of Securitas Direct Safety Radio Protocol.

The following layers are what construct the so called radio stack for our coming test application. The stack interacts directly with the application through Radio Stack API and to the platform dependent radio handler.

- **Physical Layer Radio driver** – As mentioned earlier, this provides a radio driver transmitter which basically consists of functionality basic radio packet transmission and reception.
- **Data link** – This layer provides communication between two nodes including protection which involves message authentication and encryption.
- **Transport** – The transport layer provides a protocol multiplexing service.
- **Network control** – The network control layer provides services such as login, logout, heartbeat and time synchronization. An application interacts indirectly with this layer for network management.
- **Application control** – Provides additional services such as application message reception and transmission. An application interacts a lot with this layer indirectly.

- **Application** - Provides the layer responsible for handling the payload of the different messages.

The above outlined stack acts as a central component under a set of different roles which are the following:

- **Facade** - In this mode the application communicates with the radio stack instead of with the different layers directly.
- **Dependency injector** - This provides the connection between the layers so the buffers of data can be passed in between.
- **Controller** - It acts as a controller that responds to control commands from an application and directs it to corresponding layers. It also controls layers based on the different events such as login event that lies under the application control layer.

3.5.5 Physical layer

The system relies on a 2-GFSK modulation technique in the physical layer. Conventional frequency shift keying encodes a series of data of frequency variations in a carrier. The current implementation runs at a bitrate of 38.4 kbps. There are currently two different frequency ranges in which both are regulated by European Committee for Standardization.

3.5.6 Data Link Layer

The data link layer is responsible for the communication between two end nodes. Moreover it also provides two different message types; normal and session messages which are the only types of messages that the radio traffic consists of:

- **Normal** - These types of messages are sent to and from nodes.
- **Session** - Session messages are used for sending larger amount of data. Sessions minimize protocol overhead and are used for streaming data (audio & video) and thus increasing bandwidth efficiency. The data can consist of either audio or pictures.

The data link layer ensures that the message is only sent once to the subscriber and may in actual cases where a retransmission takes place see the message more than once. Since every message that is sent has a sequence number that is kept in case of a retransmission, it enables the sequence number to stay the same for the next transmission. When the requested encryption is performed in this layer, any received message with already predefined encryption is decrypted and presented in clear text to the above layers.

3.5.7 Transport layer

The transport layer is responsible for multiplexing messages which results in transport of application messages of various types. An application can subscribe to a number of message types and it is up to the transport layer to distinguish the corresponding application of a certain message type.

3.5.7.1 Listen Before Talk

Listen Before Talk (LBT) is a principle regulated by EN 300 220 standard[8]. It can be seen as a derivative from the access method CSMA/CA specified in the IEEE 802 standard, see Figure 3.4. It refers to the process that a transmitter must perform when about to transmit and what is expected after transmission. The steps are as follows:

1. Receiver is turned on.
2. Listen on frequency for five consecutive milliseconds. If energy, i.e a carrier signal is detected on the frequency the transmission must cease.
3. Listen for a random time between 0 and 5 ms. If the presence of a carrier signal the transmission is ceased.
4. However if no energy is detected on the frequency, transmission is taken place.
5. When transmission phase has been completed the next time slot for transmission appears in 100 ms.

The procedure however does not apply to acknowledgment messages as these act as receipt of already sent messages and may therefore be sent whenever without any obligations towards LBT.

In case the transmission was ceased due to the occurrence of an existing carrier signal the transmitter must wait a random period of between 20 and 30 ms before it can enter the LBT procedure again. It is required for the node to retry a minimum of 1.5 seconds and maximum 5 seconds for it to notify upper layers in the stack in case of failure.

3.5.7.2 Listen After Talk

Upon each reception of a message, including acknowledgment messages the unit may optionally listen for further incoming packets. This is referred to Listen After Talk and its purpose is to give the unit the choice to listen for further packets

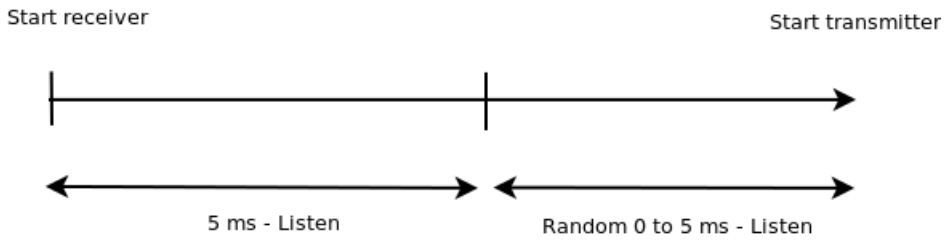

Figure 3.4: Listen Before Talk illustration

coming from non battery operated nodes. The size of the windows is usually 100 ms long and is initiated upon completion of the transmission.

In order to setup a LAT period, a bit in the header of the message must be set. However due to different product requirements messages and nodes may differ in their implementation on this matter. There is an option for example to set a sleep bit in a data link header to provide battery saving functionality and therefor avoiding LAT obligation.

3.5.8 Heartbeat Sequence

The following describes how a heartbeat interaction sequence is performed between RF master and node.

1. The Node transmits a `'nwkHeartbeat'` with destination xx (id of RF Master). The message carries status bits about battery, alarm, etc. Last protection sequence number +1 is used.

2. RF Master receives the message and responds with a `'nwkHeartbeatResponse'` with the destination field set to the node. The node will remain active on its radio receiver to be able to receive this message. The parameter stayUp =XXX notifies the node to continue to listen for more radio messages for an additional XX seconds.

3. RF Master can now send any message to the node within the given XX seconds. After the XX seconds, node will return to sleep and power-down its radio.

3.5.9 Network Control

In this layer network control functions are provided which control the different network class messages. The network class messages include functionality which

involves the login and logout procedures and the encryption handling. Time synchronization for the preservation of battery power and repeating functionality is also implemented although it is optional. The only minimum requirement required by the nodes within the network is the login.

3.5.10 Application Control

Responsible for resending handling and acknowledgement mechanisms.

3.5.11 Application

Handles the resulting payload field and decides what do based upon it.

Chapter 4

Medical Interoperability standards

There have been numerous projects in the recent years focusing on developing medical equipment operating in homes of patients. Most of the projects had focused on the optimization of the sensor networks. In 2007, the International Organization for Standardization joined the IEEE 11073 standard and later the ISO/IEEE 11073 Personal Home Data was created. The standard focuses on the interoperability between health devices making it possible for different vendors and manufacturers to co-exist in the homes of patients. It can also prove to be beneficial for the manufacturer when changing market to another country or even municipality.

The Vinnova research project itACiH currently has no focus on standards, however the proposed system architecture of itACiH can be interpreted as 11073. Making the sensor network ISO/IEEE 11073 PHD compliant could create business opportunities in other regions and countries then the Skåne region.

4.1 ISO/IEEE 11073 Personal Home Data

In 2004 the standard ISO/IEEE 11073 was released and described criteria for how medical equipment should communicate with each other. This standard was mainly focused towards the hospitals, rather than the homes of patients. Therefore, the standard was developed further to take into account the growing population, which will require home care.

Concerns were raised that the roll-out of such home systems will take a lot of time because the systems available would be incompatible with each other, which is why the ISO/IEEE 11073 Personal Home Data (PHD) was developed. The intention was to allow for individuals to keep track of their current health status through different services and even to have professional care takers to monitor their status, see Figure 4.1 for the focus area.

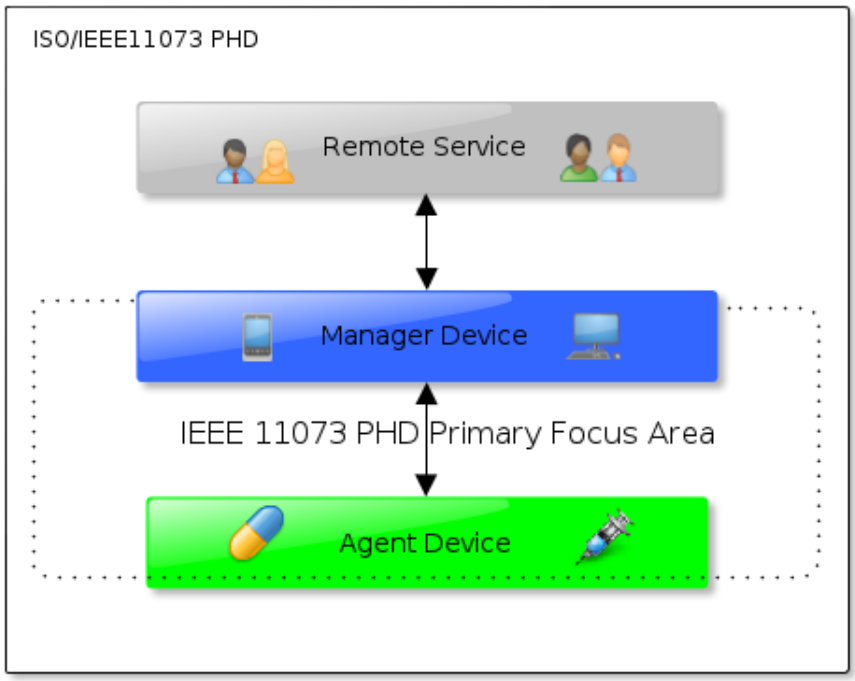

Figure 4.1: ISO/IEEE 11073 PHD Primary Focus Area

ISO/IEEE 11073 PHD standards are all built upon the framework standard IEEE Std 11073-20601. This standard describes the application profile, how data should be formatted in a generic way, what type of messages should these devices communicate with and the communication model. Other standards only describe how new, type specific standards should act and communicate. Both the framework standards and the device specialization standards are listed in table 4.1 and 4.2. The protocol stack is made up from these standards, as shown in Figure 4.3.

As described by the standard IEEE 11073-20601-2008 *"...addresses a need for an openly defined, independent standard for converting the information profile [of personal health devices] into an interoperable transmission format so the information can be exchanged to and from personal telehealth devices and compute engines (e.g., cell phones, personal computers, personal health appliances, and set top boxes)"*[6].

Table 4.1: Framework standards

Name	Type
IEEE Std 11073-20601	Application profile
IEEE Std 11073-20601a	Application profile (amendment)

Table 4.2: Device specialization standards

Name	Type
IEEE Std 11073-10404	Pulse Oximeter
IEEE Std 11073-10407	Blood Pressure Monitor
IEEE Std 11073-10408	Thermometer
IEEE Std 11073-10415	Weighing Scale
IEEE Std 11073-10417	Glucose Meter
IEEE Std 11073-10420	Body composition analyzer
IEEE Std 11073-10421	Peak flow
IEEE Std 11073-10441	Cardiovascular fitness and activity monitor
IEEE Std 11073-10442	Strength fitness equipment
IEEE Std 11073-10471	Independent living activity hub
IEEE Std 11073-10472	Medication monitor

4.1.1 System model

The system model is divided into three main components, which are treated separately in IEEE 11073-20601. It contains a Domain Information Model (DIM), a Service Model (SM) and the communication model (CM), as described by Figure 4.2. The different parts are handled separately in the PHD standard in order for manufacturers to achieve interoperability between the different products[12].

4.1.1.1 Domain Information Model

The DIM is a model focused on the objects. It defines all the different devices in form of classes, which are able to connect to the standard in an object oriented way. Each object has more than one attribute and each attribute in the objects describes different states of the device.

The model describes an agent device as a set of objects that represent the data sources. These data sources are treated as elements that the manager can control the behavior of. This can be performed by different methods such a defined SET and GET methods. The standard describes what actions should be taken when a SET or GET method is used on a a specific device.

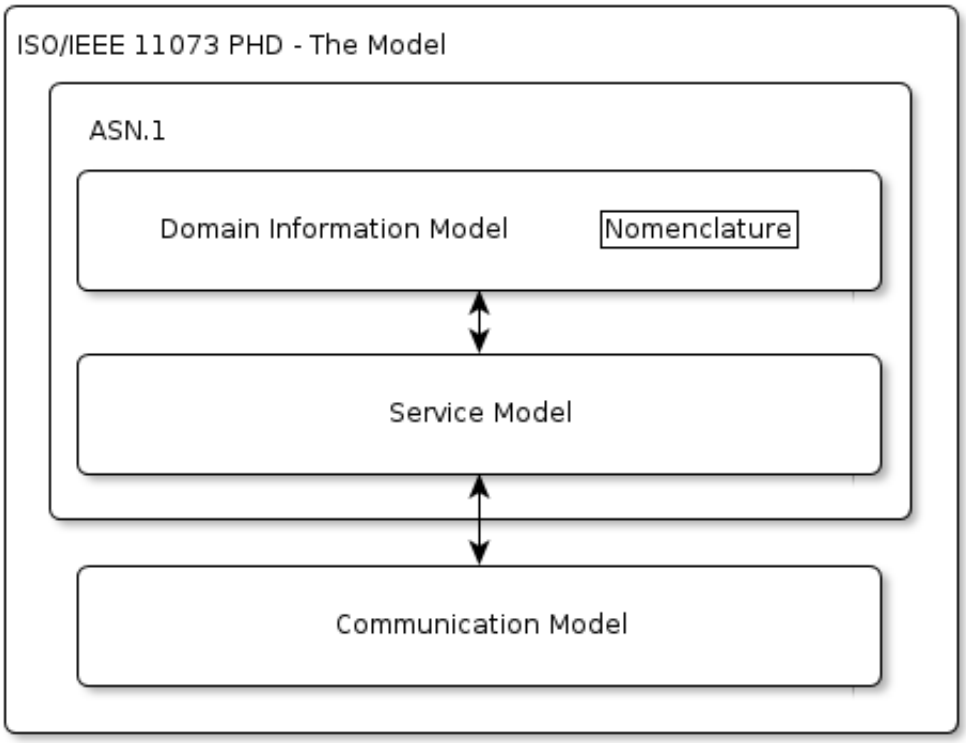

Figure 4.2: The ISO/IEEE 11073 PHD Model

Regarding to IEEE Standard document[6] the object oriented information model supports the following.

- Separation of specification from implementation through the principle of encapsulation.
- Support for evolution through the principle of inheritance.
- Support for backward compatibility through the principle of polymorphism.

4.1.1.2 Service model

The service model defines the mechanism of data exchange services between the agent and the manager. These services are mapped to messages that are exchanged between the agent and manager. All of the protocol messages in the ISO/IEEE 11073 family are defined in Abstract Syntax Notation One (ASN.1). This means that messages can coexist with other messages defined in earlier versions of ISO/IEEE 11073 (not PHD) if they are defined in ASN.1.

The service model also includes definitions on how the protocol messages should be structured. The Association service should handle the communication between the agent and manager, for example, an Association request is made through the upper layers of the connection (to be able to have *any* transport layer). The document continues to describe the association service with other parameters such as a response to the earlier described request, as well as a Release of the request, a response to the release and a abort message that terminates the upper layer associations immediately without a response.

The service model part continues to describe the object access services with different access services that are supported, such as a GET and SET service, so the manager can change different parameters of the agent.

4.1.1.3 Communication model

The standard bears in mind that the requirements are based on the fact that people should have these kind of techniques in the home, often in mobile environments. The standard assumes that the transport technologies (currently Bluetooth-LE, ZigBee® (IEEE 802.15.4) and USB) are feature rich and takes a generic view of transport technologies[3].

The standard describes three different types of transport profiles[6].

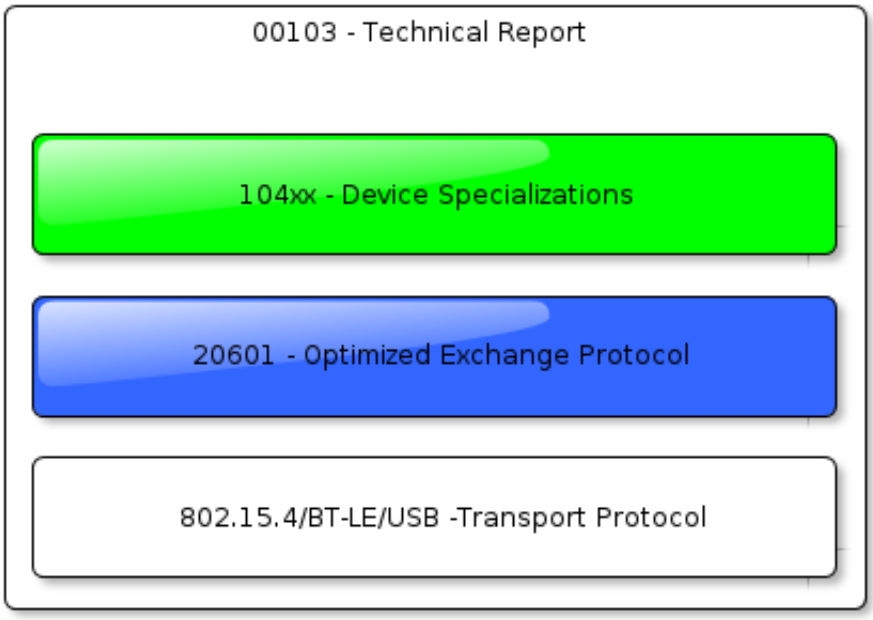

Figure 4.3: The IEEE 11073 PHD Protocol Stack

- **Type 1:** A transport profile that contain both a reliable stream (with resending) and a normal datagram stream (best-effort).
- **Type 2:** A transport profile that only contains an unidirectional transport service.
- **Type 3:** A transport profile that only contain a best-effort transport service.

The core feature of the communication model is to define the correspondence of the network in a one-to-one connection between the agent and the manager[3]. The connection is handled with the finite state machine, see Figure 4.4. Also to maintain compatibility between devices, old or new as well as code data from the DIM. The state machine described in Figure 4.4 describes the different ways of communications between the states, however the exact attributes associated with the different tasks are excluded in the diagram.

4.1.1.4 Agent State Description

The different states described in Figure 4.4 are:

- **Unassociated:** The agent is in the Unassociated state when it does not have an application layer association with a manager. This occurs when a new connection was established, the manager rejects a request of association or the agent or manager releases or aborts an active association.
- **Associating:** After determining if it should create an association, the agent moves to the Associating state and sends a request to the manager. If it fails, however alternative associations are possible, the agent may attempt to the association once again. If timeout occurs, it shall attempt to associate to the maximum retry count defined.
- **Associated:** When the manager accepts the association and have checked that they share common versions and protocols, it sends an accepted parameter.
- **Operating:** After the accepted parameter has been sent from the "Associated state", it will be in the "Operating state". Hence, the devices are now associated and one can acquire the information.
- **Configuring:** If a manager does not recognize the agents configuration parameters, it will ask the agent to send an "Association Response" with a parameter called: "accepted-unknown-config". This parameter indicates that the association has been accepted but the configuration needs to be

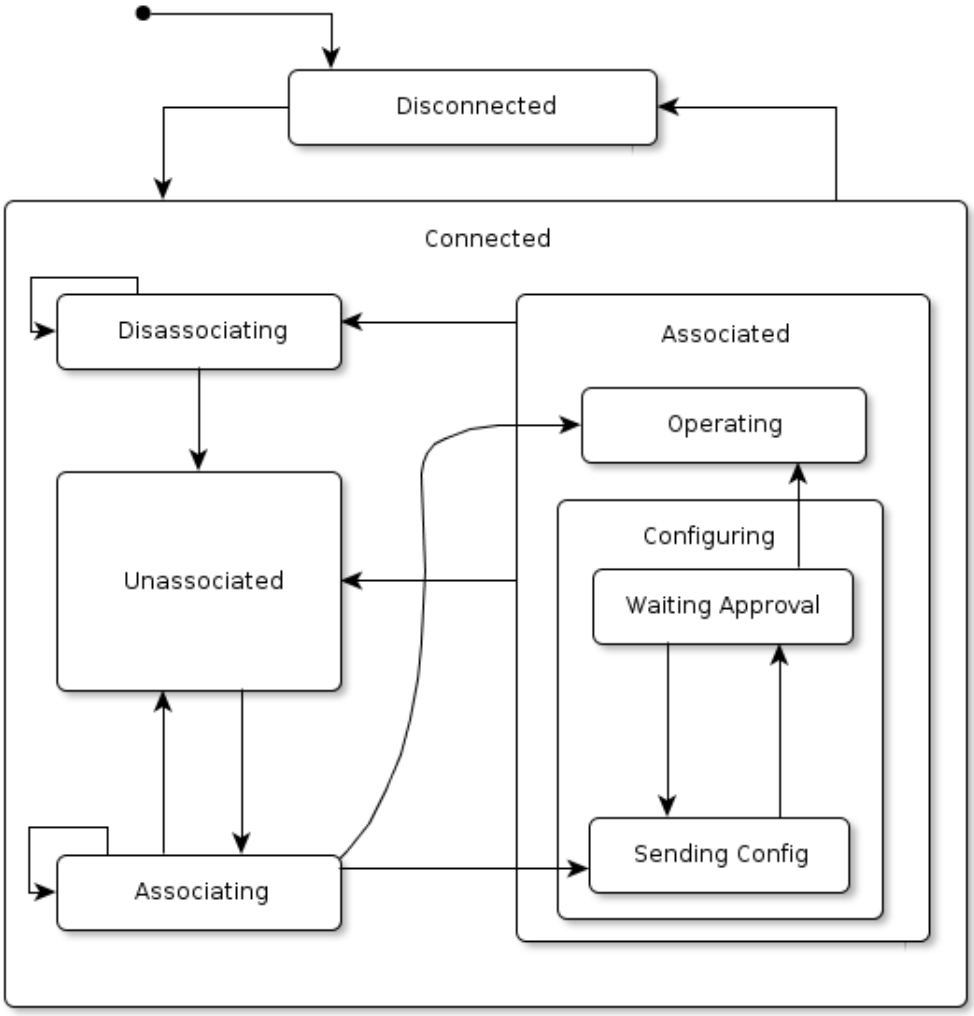

Figure 4.4: The Agent State Machine Diagram

transmitted. The agent stays in this state until the manager acknowledges the unknown configuration.

- **Disassociating:** When an agent determines that it should release the current association, the agent moves to the Disassociation state and sends a request for releasing the current association.

The states in Figure 4.5 are very much like the ones from Figure 4.4. The differences are:

- The Manager must wait for the "Waiting for Config state" for at least 10

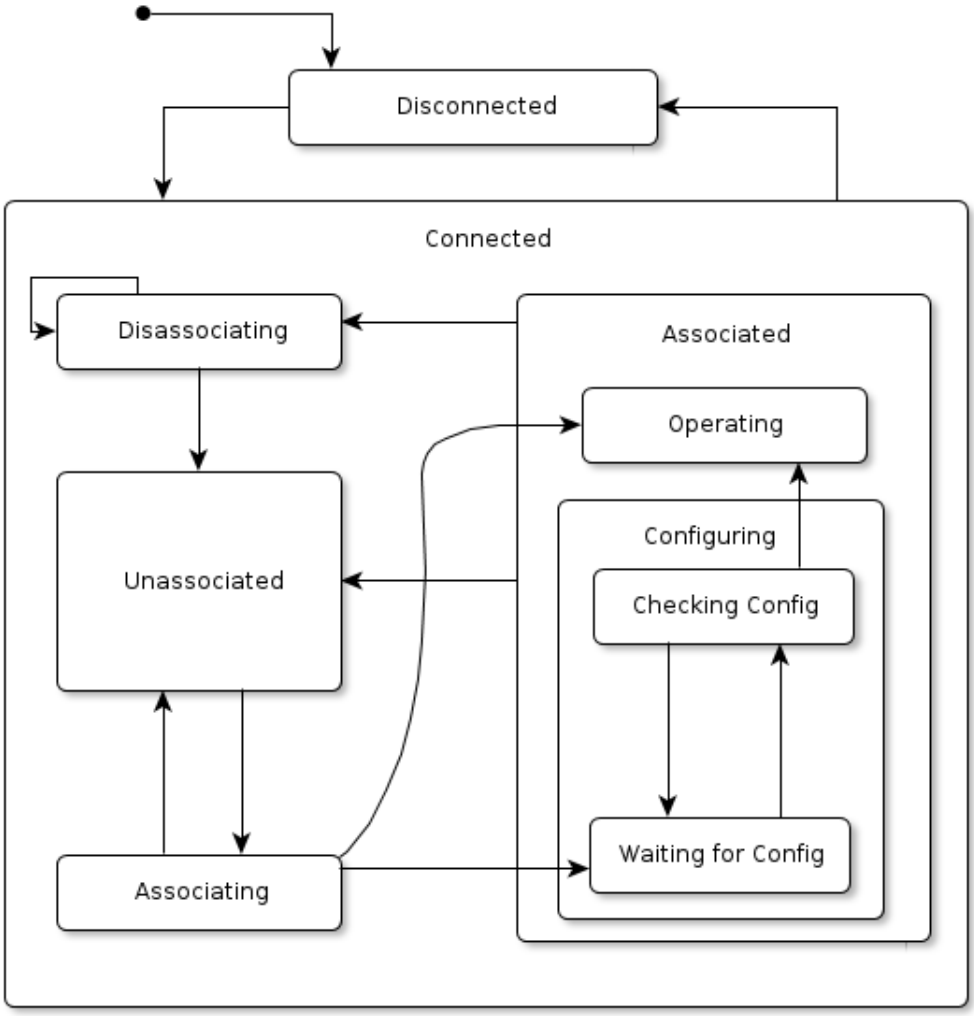

Figure 4.5: The Manager State Machine Diagram

seconds before the agent sends its Assosiation Release Request or Assosiation Abort message.

- If a manager does not support a new configuration, it should send back a response with an unsupported-config result and if accepted, it must send a configuration response with an accepted config result.

This description of the different states illuminates a brief understanding on the state machine, however, for further information regarding the transition parameters see [6]. This standard, together with the device specific standards give a good understanding on how what attributes and parameters that should be sent and received.

4.2 The Continua Health Alliance

The Continua Health Alliance (CHA) is an alliance created to realize the standard ISO/IEEE 11073. CHA is a non-profit, open industry alliance with members from the leading health care and technology companies. The alliance was created in order to improve the interoperability between products to establish a health system that will improve the health and wellness of patients[2]. The Alliance have a set of guidelines and principles that companies should follow in order to receive a certification. The terminology of the guidelines differ a bit from the ISO/IEEE 11073 standard.

The different parts of the system are PAN/LAN Device, PAN/LAN-Interface, Application Hosting Device, WAN-Interface, WAN Device, xHRN-Interface and Health Record Device.

4.2.1 PAN/LAN Device

PAN, Personal Area Network Device or Local Area Network Device, is what ISO/IEEE 11073 refers to as the agent device. It could be one of the device specializations listed in Table 4.2. This device should have the ability to communicate with the Application Hosting Device. A LAN device is also mentioned in the Continua Health Alliance Guidelines, it works in the same manner but with a LAN interface that connects to the Application Hosting Device.

4.2.2 PAN/LAN-Interface

The PAN-interface or LAN-interface is the interface that connect the PAN/LAN-devcies with the Application Hosting Device. It enables the PAN-Devices to connect to any Application Hosting Device, of any vendor.

4.2.3 Application Hosting Device

The Application Hosting Device is referred to the Manager in ISO/IEEE 11073. This is the central point of control in form of a cellphone, computer, set-top box or a hub. The manager device should be connected to the WAN device.

4.2.4 WAN-Interface

The WAN Interface defines the interface between the Application Hosting Device to the WAN-Device. This is usually the link between the home or office to the back end service provider.

4.2.5 WAN Device

The device stores patient information. The guidelines do not define how the Application Hosting Device should communicate with the WAN device. However the WAN device may forward the data to a care center over the Health Record Network (xHRN) interface using a standardized way to represent health data called HL7. It could also be a middle hand that passes through all the data collected in a secure way. For instance, to be able to have confidentiality on the data, integrity on the transmitted data and non-repudiation on the data transactions in order to make a secure way for sensitive data to pass through systems. One need to be able to assure availability on the data is also crucial (eg. DDoS attacks).

4.2.6 xHRN-Interface

This interface describes the connection between the back end service (WAN-Device) and makes sure that the data is passed on to the Health Record Device. Traditionally, patients have not been able to see their own health record, instead it has been exclusive for doctors to access the electronic repository of the health records. The standard addresses this issue and makes it feasible for patients to access and keep track of their own health record. It is called a Personal Health Record (PHR), where xHRN interfaces makes it feasible for patients to store their own data in their PHR and at the same time pass the data onto the health center[18].

4.2.7 Health Record Device

The Health Record Device (HRD) is for example a service from a local health center, municipal hospital or fitness centers that store information about weight, diet and so on. The HRD should have contact with the WAN either by direct contact with the Application Hosting Device or through the third part that is responsible for the data transmission.

Between the four devices there are interfaces that are defined (see Figure 4.6). It is actually these interfaces that were defined in the first version of The Continua Guidelines. The first version focused on the PAN-interface (the interface between the PAN-Device and the Application Hosting Device) and the xHRN-interface

focused on the connection between the WAN-Device and the Health Record Device[18]. The main focus of The Continua Health Alliance is to provide interoperability between different manufacturers. It focuses on two points to acquire the benefits by interoperability[2]:

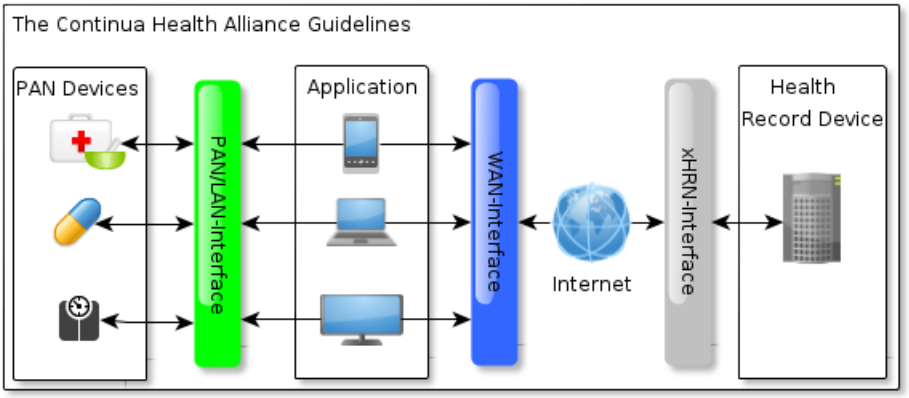

Figure 4.6: The Continua Health Alliance Guidelines

- Identifying a reference topology with well-defined device classes and interface based on domain driven use cases.
- Establishing open standards derived from existing ones in order to clearly define the interfaces.

Having an alliance that realizes a standard reveals several business opportunities for a range of manufacturers and companies. For example consumer electronics like pulse meters for fitness purposes, weighting scales or other medical monitoring devices, as well as extensive competition between ISPs, Home Security Companies and even Hospitals and Health Care Units[2].

The modules that should incorporate with the standard can already be based on the older IEEE 11073 standard (used for Hospitals) meaning that the device manufacturer does not have to be involved in the development process of the reliable communication software which is expensive. Furthermore tells the guidelines that the monitoring software or devices should not have to be proprietary since the data is handled in a standardized manner. By passing the Continua Certification medical sensor manufacturers can assure interoperability with a certified monitoring device.

Chapter 5

Results

This chapter presents the results gained. It will present an implementation of a proprietary based medical application that measure the levels of hemoglobin in the blood of patients. By performing an implementation approach, knowledge in how well the proprietary based protocol, provided by Securitas Direct AB (Verisure) can fulfill the requirements of remote medical application can be obtained. In addition, an comparison will be provided to determine how well an industry accepted protocol like ZigBe^{TM}stands against an open protocol such as SimpliciTI^{TM}. A proposition to different changes to the medical system will be presented and a final conclusive section regarding the problems with the proposal is given.

5.1 Implementation and Development

A designer or manufacturer holds the choice to select a protocol which consists of a different number of layers implemented. In this sense one can way customize remaining layers to suit his/her application needs. Current networking implementations do not fully implement all seven layers of the OSI-model and in some cases layers are interleaved or mixed with each other to better suit the application, hence the different outlines of the protocol may vary. Since a proprietary based solution will be used, slight modifications differ from a standard OSI stack in its default outline due to the reasons discussed previously.

To understand the system of the company a test implementation was performed to see if it was possible to send medical data. The system is a proprietary based solution and what is required here is to provide an implementation or modification in the application layer without thinking about the lower operating layers. By doing this we can avoid increasing the complexity of our application and maintain already optimized communication efficiency.

Since there is no way to create a reliable data-stream e.g. a TCP connection

with the Gateway, one needs to find other solutions. In our case we found out that doing a REST call with three bytes is the only way to contact the node with a packet loaded with three bytes of disguised data. The protocol acts as if a configuration to a keypad is being sent, in this case, decimal encoded ASCII letters.

On the other hand it is simpler to send up data. The protocol has a built in command that sends a fault number loaded with four bytes of data and it is able to send these messages fast through the sensor network and to the database. The database can then read the decimal encoded four bytes and chunk out the required information. The same database table is used to see if the REST calls have been received, this is handled by the one-sided resending algorithm.

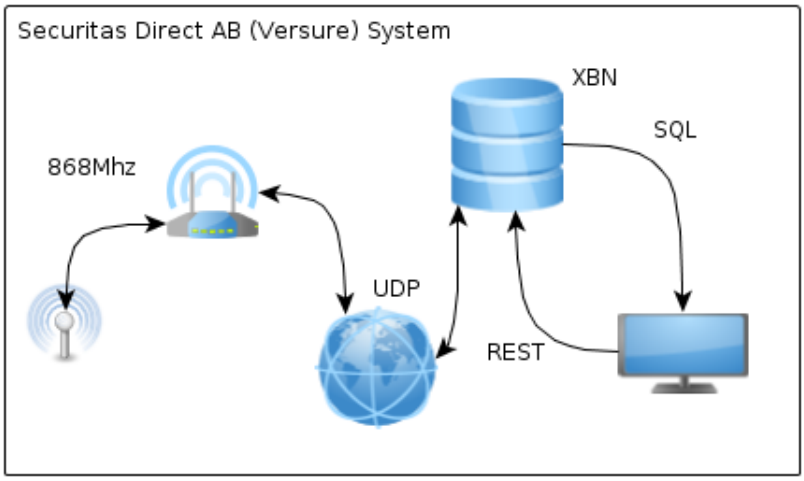

Figure 5.1: The System architecture of Securitas Direct AB (Verisure)

5.1.1 Hardware

This section will describe the hardware. It consists of a radio, microcontroller, a test board and a system on chip.

5.1.1.1 Radio

Texas instruments provide five different families of TI radios. One of them include the following radios *CC2510, CC2511, CC1110, CC1111.*

5.1.1.2 Microcontroller

Texas instruments provide two families of microcontrollers : Intel 8051 and TI MSP430. The former is the one used in our implementation.

5.1.1.3 Board

A number of boards are supported, all with different functionality. In our case the SRF04EB will be used.

5.1.1.4 CC1110EM

The CC1110 868-915 MHz evaluation modules are add-on daughter boards that require SmartRF04DKs for evaluation and development. The CC1110 Evaluation Module is a small plug-in module for SmartRF04EB, and is used as reference design for RF layout. The application is currently running with the magnet contact software from the SD client stack (Rev.1215) with some minor changes. In order to receive packages intended for a keyad (e.g packet keypad beep), we had to change the device type in the software allowing the application node to be identified as a keypad in our network, thus making it possible to receive config settings of larger size by simple SD-API requests. This is done by modifying the sd_radio_info.c class.

```
const uint8_t sd_radio_info_device_type = SD_RADIO_NWK_DEVICE_TYPE_KEYPAD;
```

5.1.2 Early implementation

In order to send a test ASCII message from the current testboard we had to perform a violation event on the node, in our case a simulated event was triggered describing a magnet contact that had been removed. The implementation consisted of checking what kind of event that had occurred (e.g violation of a contact). This generated an `'appViolation'` message consisting of two bytes that was sent to the gateway. The gateway acknowledges this message with an `'appViolation'` acknowledgment consisting of one byte payload. An `'appFaultNotification'` consisting of four bytes was then sent as a response to the acknowledgment which was enough for us to send our current test message `'HEJ!'` to the gateway. The four bytes message is then added to a table called node_faults in the database containing the a set of logs regarding our device.

5.1.3 Default implementation

The default implementation sends a heartbeat every minute to ensure the gateway that the device is alive. However, here we have decided to send a heartbeat every second which allows us to enter the LAT state and check if there is anything of importance for the application node to receive. Since we lack any NTM (Network Time Management) logic e.g time synchronization in the implementation which would allow the gateway to wake up our device we have chosen to send frequent heartbeats.

5.1.3.1 UART

All data transmitted to the UART (Universal Asynchronous Receiver/Transmitter) interface on the CC1110EM will be sent to the database node_fault table and security precautions will be taken in later revisions. In order to setup a serial connection to the application node certain parameters must be adjusted. The following set of parameters are required. A baud rate of 9600 kpbs, 8 databits, one stop bit and no parity or flow control is set. The intended computer to run a terminal should also have a USB to serial driver installed in order to connect through the serial port. Terminal clients such as PuTTy or GTKTerm are recommended.

5.1.4 Communication on application node

Since we discovered that the sensor network of Securitas Direct AB (Verisure) is not able to send and receive at the same time, or via the same type of technique, the sending and receiving mechanisms are separated.

5.1.4.1 Transmitting

Before every heartbeat a routine is done to check if we have received anything on the UART-interface. When one symbol has been received, in this case on the UART interface, the symbols are shifted eight bits and exclusively or:ed. This task is performed until a total of four symbols have been received which corresponds to a 32 bit integer. This integer is then added to a node fault log which is handled by the default implementation by transmitting the node fault message to the gateway. The application node normally transmits the `'appFaultNotification'` status message consisting of four bytes to the gateway upon detection of a system fault condition. In our case a a generated ”false” status message containing ASCII symbols.

5.1.4.2 Receiving

When a packet is received from the gateway the different fields in payload are checked to see whether the payload is a legitimate payload or not. The current application allows a the node to receive `'AppSetProperty.propConfig'`. The packet that has three different parameters,`'Frequency'` , `'Duration'` and `'Volume'`. The gateway transmits the `'appSetProperty.propConfig.configKeyBeep'` configuration message to the node in order to change the key press beep pattern. Each field is one byte which is suitable for masking in a ASCII symbol and in our case masking a message consisting of three bytes. Each received message is acknowledged to the gateway.

One can improve message delivery by introducing acknowledgements schemes, which comprises of the receving node sending an acknowledgment packet upon reception to the original sender to confirm successful reception of the corresponding packet. By using a predefined number of retries the probability of experiencing packet transaction loss can be greatly reduced. That is why we introduce a form of ACK schema in our implementation.

5.2 The front end

The front end of the application currently runs in Java with a GUI using SWT (this makes it possible to run a interface on all major platforms, 32 and 64 bit). This application can not run off site, since it needs to access the database, which only allow connections from inside the intranet. At the moment, no tests have been done on the functionality of this working outside the site.

5.2.1 GUI

The application has two views; one basic view (Figure 5.2) and one advanced view (Figure 5.3). The basic view has three buttons, one text field and three radio buttons. The three buttons determine what action you want to perform.

If you press "Get new bloodvalue" the front end will send a command to the CC1110. The latest measured value is then obtained from the memory of the Hemocue device and the data obtained from the Hemocue device is then sent to the database. The front end will then gather the latest data measured and display it in the text field. If some errors occur, the user is free to change the display option. The normal way to represent the data from the Hemocue device is to encode the data in base64-lite. If the input looks corrupted, the user can change the way one desire to represent the data from the database in ASCII format, or in HEX format.

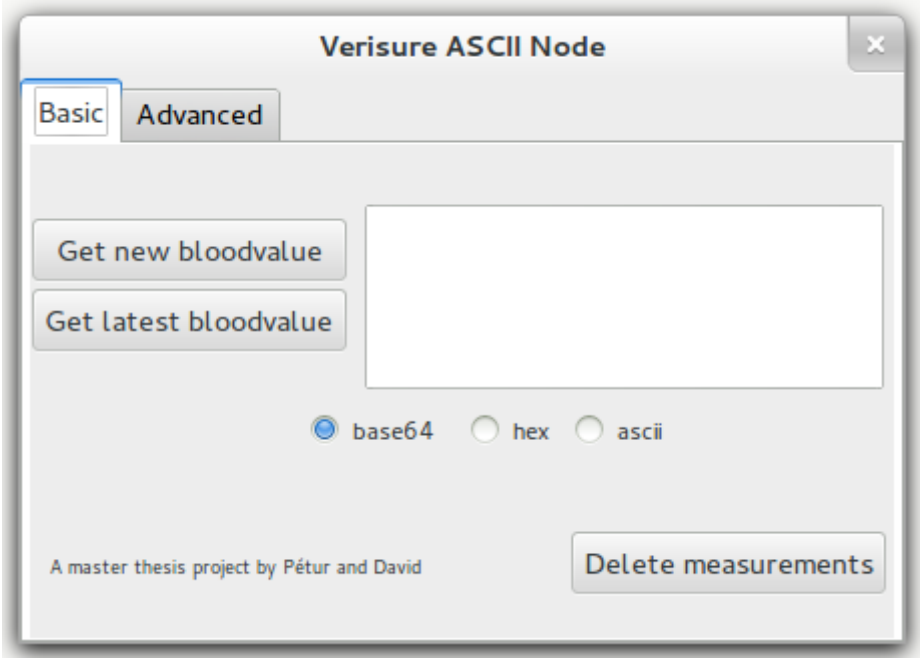

Figure 5.2: Screenshot of the Basic view

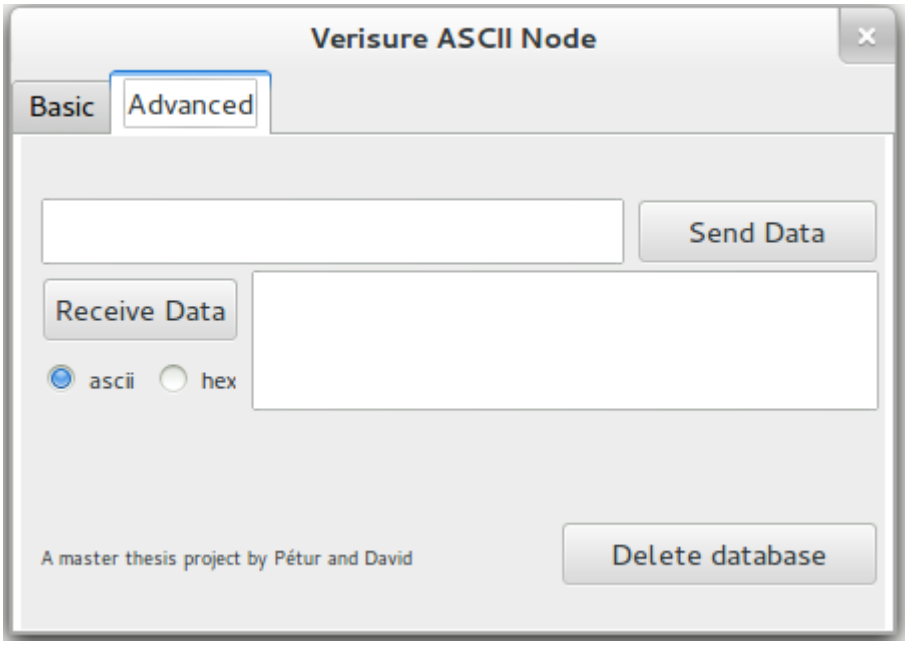

Figure 5.3: Screenshot of the Advanced view

The button "Get latest bloodvalue" will query the database, without sending any data to the CC1110. This will get the latest value that was triggered by the "Get new bloodvalue" command. This button can also use the three radio buttons.

The "Delete measurements button" will erase the measured data from the Hemocue device. It has a memory which stores more than 500 values, if one want to delete those values, press the "Delete measurements" button.

The "Get new bloodvalue" will trigger a REST call that is sent to the XBN(see Figure 5.1), the function used is:

```
PUT /installation/cid/config HTTP/1.1
```

Three parameters are sent, as described below.

```
<request>
  <installationConfig>
    <cid>00005778</cid>
    <keypadBeepFreq>76</keypadBeepFreq>
 <keypadBeepDuration>79</keypadBeepDuration>
    <keypadBeepVolume>71</keypadBeepVolume>
    </installationConfig>
</request>
```

The Send function will send three bytes masked as decimal ASCII values in the three parameters. A HTTP client class is used to be able to send these parameters. The Send function also uses a resending mechanism. It sends the three bytes to the CC1110, then waits for the CC1110 to send a node fault message with the value `'ACKS'` to the database. After three seconds, the send function will see if it has received an `'ACKS'` from the device, if so, it continues to send the next three bytes (only used in the advanced view), if not, it first sends a dummy packet with the value ÉÉÉ to erase the current value of the database (it does not accept duplicate packages twice in a row), which is ignored on the CC1110, and after 500 ms it resends the packet, waits an additional three seconds and then checks for an ACKS in the database.

This is a very straightforward, one sided, resending technique, but since we cannot implement the resending functionality on the CC1110, we can never guarantee that the packet has not already arrived. If a database check is done with the back end and no ACKS is present, another request will sent again. However, the packet can be on the way from the CC1110. It can be guaranteed that no packet has arrived if the the sleep time increases between sending and database check with a longer time then three seconds, however a great loss in performance will be noticed.

The Receive area is in a way much simpler, since all the functionality is implemented in the client stack of the CC1110 and the gateway. The receive area reads the database with desired cid (an installation id) and parses through the whole `node_faults` column which is in decimal form. The back end then parses through the decimal data, changes it to hexadecimal form, parses the hexadecimal numbers to String format and then does a char conversion of the hexadecimal String format and ignores the packets with the value `'ACKS'`, which is used by the resending mechanism.

5.2.1.1 Advanced view

The Advanced view lets the user send arbitrary data from the front end to our medical node. The current release of the application code does not support any other inputs than "LOG" or "DEL". However, if the Send area has more than three bytes, the send mechanism mentioned earlier will chunk the data to three bytes at a time. The resend mechanism works in the same way, it doesn't send the next three bytes until the last packet was acknowledged.

The view has also a Receive window, which actually works in the same way as the Basic tab, with the exception that it cannot display the bloodvalue encoded in base64-lite.

The Advanced view also has a Delete button, this button does not Delete the measurements from the Hemocue device. It deletes the entries in the database with the given installation id.

5.2.2 A Hemoglobin measurement device

To our help we have a hemoglobin measurement device that can send different values from a log through a serial connector (UART). The convention is to send the value from the first log position. The packet format data is defined by its own in the document: Description of the HemoCue 201+ Communication Protocol (FSF0210/1)[7]. Here we had to do a base64lite conversion on the data sent. The device is able to receive 11 bytes of predefined packets and send 13 bytes.

All the data sent to the UART interface is sent with the `node_fault` packet, so for example both the start flag (0x82) and the end flag (0x83) is send through the `node_fault`. Then the measurement data is three bytes encoded with base64lite and parsed once again with the back end, and later displayed on the Receive window screen. First, an event has to be triggered with the command 'LOG' in the Send area. The structure below is how a regular log request is formatted.

Table 5.1: Request log packet

<STX >	1 byte	Start character(0x82)
'L'	1 byte	Command
NN	2 bytes	The memory position is coded in base64.
	5 bytes	Padding
	1 byte	Checksum
<ETX >	1 byte	End character(0x83)

5.2.3 Usage

This application is dependent of three parts, one Hemoglobin meter (HemoCue Hb 201+) one CC1110 on a SmartRF04Eb Evaluation Board that is connected through a serial connector through the UART interface and a front end.

First, the user must do a measurement in according to the HemoCue Hb 201+ manual. After the sample is put into the tray of the measurement device the value will be available on the screen on the HemoCue device.

The usage of application from the Basic tab should be very straight forward. To obtain a new blood value, press the button: "Get new bloodvalue". This will trigger the command LOG, desribed in the 'LOG' flowchart, as shown by Figure 5.4. The Figure 5.5 shows what happens when DEL is sent, it also illustrates a retransmission.

5.2.4 System Requirements

The application has been tested on the following machines:

- Lenovo ThinkPad T430, Intel Core i5 and 4GB RAM
- Linux: OpenJDK 1.7.x
- Windows: JSE 1.7.x

The application is built with Standard Widget Toolkit, SWT, and needs an extra JAR in the compilation process.

5.3 Protocol analysis

The different results from the protocol analysis as well as the analysis of future implementation of the ISO/IEEE 11073 standard will be discussed in this chapter. We will compare the SimpliciTI™ based standard together with the ZigBee® (IEEE 802.15.4) standard in order to give a view of what to consider when choosing in between an open protocol and an industry accepted standard such as ZigBee.

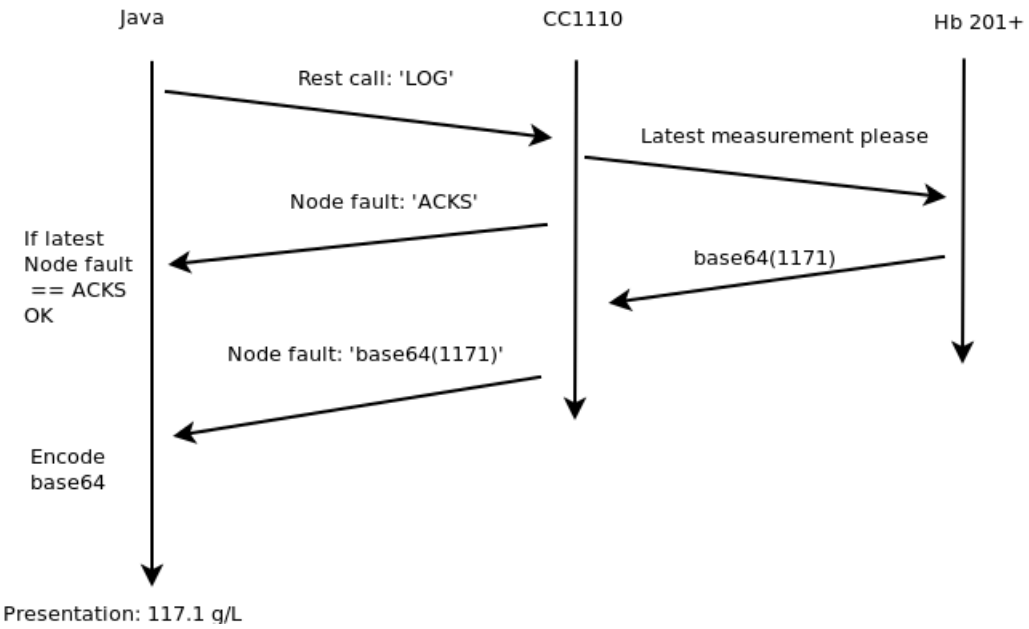

Figure 5.4: Flowchart of the use case 'LOG'. This flowchart describes the case when pressing the Delete measurements.

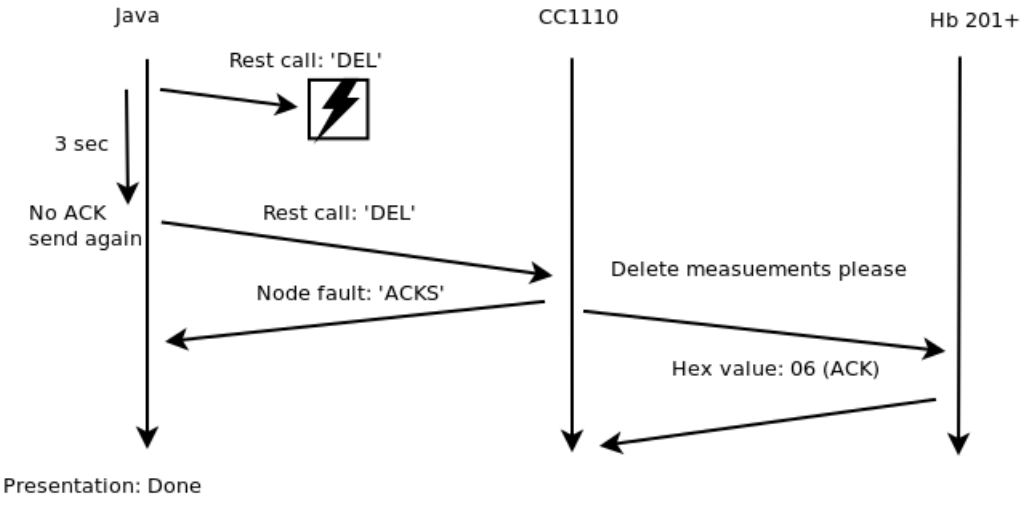

Figure 5.5: Flowchart of the use case 'DEL' with retransmission. To obtain the latest measurement stored in the database the button "Get latest bloodvalue" is pressed.

5.3.1 SimpliciTI™ and ZigBee® Comparison

A set of factors are to be considered when standing in the choice of using a suitable protocol suite for the application. The initial step in determining which choice of protocol is to define the different application criteria. It is required to reflect regarding the network parameters such as topology, packet size and critical reception in terms of reliability and security. Additionally, it should be discussed how well the protocol can be customized to be able to be applied to the application and if so how flexible it is to do so. It should also be considered how much time is estimated to be spent on learning and adjusting the protocol to the current application and determine whether it would be more feasible to develop a more customized solution.

It can be noted that the balance between functionality and simplicity of the IEEE 802.15.4 lower layer gives the opportunity to use already existing software. In addition with an already implemented task scheduler provided with the two underlying lower layers, the cost and time spent by an engineer to provide fully covered solution is greatly reduced, especially when it comes to the reliability of the communication.

5.3.1.1 Environment

Evidently ZigBee® may be suitable in environments or for products that require trust in an already standardized physical layer and lower layer protocols(802.15.4) due to the robustness and reliability of these layers. In addition a standardized higher layer protocol is provided in the ZigBee stack allowing for networks to be configured in several hierarchy modes such as mesh topology and multihop)[5]. ZigBee also allows for complete interoperability up to the public profiles between the different layers which simplifies development which should mainly be focused on the application. The industry has undergone a larger acceptance of the protocol contributing to high support and maintenance between vendors and providers.

5.3.1.2 Costs

One setback is the cost for the ZigBee alliance membership, through a yearly membership fee paid to the ZigBee alliance and the additional cost to certify the product as ZigBee compliant e.g certification process, as well as for the memory allocation of the protocol itself.

Since the ZigBee protocol includes a range of features that may be proven to be somewhat difficult to put in use, thus setting higher strain on current memory resources. This would be memory which in a customized setting that easily could

have been put to better use. That is why in some cases memory and resource requiremens can be shown to restrict the end-application[15].

For this very reason a number of companies have decided launch radio with integrated MCUs which already include the ZigBee software. The available operations are then accessed by a small set of different API calls through a application-centric MCU.

On the other hand for a protocol such as SimpliciTI^{TM} or other existing, similar, low-level implementations, the applicational needs would be the following:

- High flexibility to design higher upper layers
- Design and development costs that must be lower than a pure proprietary solution
- Access to lower layer protocol for faster implementaion and deployment in terms of out-of-the-box

5.3.1.3 Layer handling

The SimpliciTI^{TM} protocol has a port architecture similar to the TCP/IP suite protocol used to communicate with a network layer that provides the management services. It also maintains a minimal so called Board Support Package (BSP) layer which is used to interface the radio and MCU[17].

The protocol itself has no formal physical layer description, thus no actual data rate, frequency or modulation requirements are provided which gives the designer the choice of design and outline of the protocol at the hardware level. Of course, the logical choice of protocol depends entirely on the application and SimpliciTI^{TM} 's essential features lie in its small memory allocation, ease of use, and reduced complexity.

However if accepted as protocol of choice, the developer must accept drawbacks such as being obliged to stand for implementation of higher level layers. There may also be some royalty fees involved if participating in a group of companies supporting the standard. On the other hand the full source for SimpliciTI^{TM} is distributed free of charge and royalties, although the hardware restricted to hardware from the same company.

Last but not least what mainly differs between these two protocols is the complexity. Since the wireless modules of the SimpliciTI^{TM} use basic core API it can allow for a more flexible network design approach. The given MAC-sub layer implementation requires a code size of 8-16 kbyte as opposed by the ZigBee radio stack[16]. It is therefor of high importance not only to consider a range of features but also a high readability factor in conjunction with memory usage when it comes to selecting an appropriate protocol.

Wireless protocols like ZigBee and SimpliciTI^{TM} well provide what an embedded developer can require since the protocol architecture is made accessible just sufficiently in order to be intuitively to fully take part of the different features provided.

5.4 Proposed implementations of ISO/IEEE11073 PHD

In this section we will try to describe what actions are needed in order to fulfill the standard and the Continua alliance guidelines. The system that should deliver the medicine data can be interpreted in different ways. We have to determine if we should implement the agent or the manager. Implementing both the manager and the agent is not a good idea, mostly because that the interoperability may come in second hand, considering that we work with the whole system and also in terms of development costs.

5.4.1 Proposed Securitas Direct Safety Radio System as the Agent

ISO/IEEE 11073 PHD is based on the principle that different devices; agents and managers, should be able to connect to each other over the same transport protocol, see Figure 5.6. However, since the security system is not based on any of the accepted protocols, BT-LE (Bluetooth Low Energy) or ZigBee, we have to interpret the system as a LAN-device. Hence, the whole system from the medication device, the wireless application node, the gateway with the Internet connectivity as well as the backbone of Securitas Direct AB (Verisure) have to be the agent, see Figure 5.7. On the other side of the backbone, it can communicate with either the manager, the remote service, or both.

The agent in ISO/IEEE 11073 PHD defines the device that should communicate with the manager through an interface called the PAN-interface (in the Continua Alliance). A case could be that the user has both a manager device and an agent device with the different specialization devices.

Due to the restrictions on the transport layer, realizing this principle in Securitas Direct AB (Verisure) proprietary protocol is probably not possible. Instead the whole system, from the node and the specialization device, to the gateway, throughout the Internet and to the back end needs to be interpreted as the agent. This on the other hand is not accepted by Continua, however, it might be possible to get the system to imitate the standard but not acquire any advantage from it. This system is more complex and will result in a larger development cost than if it was based on ZigBee from the beginning. Since Securitas Direct AB (Verisure) does not use accepted transport layers, the whole system has to be interpreted in a

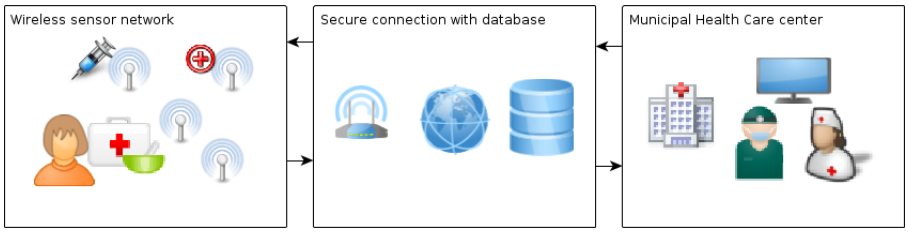

Figure 5.6: A ZigBee based system with an ISO/IEEE 11073 PHD Implementation

different way and it has to be based on joint project between a medical equipment company and Securitas Direct AB (Verisure) and together realise what should be recognized as the Agent. Regarding the back end, Securitas Direct AB (Verisure) has to implement the interface so the user can connect managers from other vendors to the system.

Figure 5.7: The Agent from IEEE 11073 PHD

5.4.2 Proposed ZigBee/BT-LE system as the Manager

A better way to implement the system in order to achive the Continua health Alliance certification is by implementing a ZigBee or BT-LE device into the gateway that would have the responsibility to communicate with agents that do not use Securitas Direct Safety Radio Protocol. Hence, the gateway should be the manager

Figure 5.8: The Manager with a BT device in the Gateway based on IEEE 11073 PHD

and Securitas Direct AB (Verisure) back end should handle the data transmission to the health centers and hospitals or other remote services, see Figure 5.8.

The whole implementation would look more like the Continua Alliance guidelines if ZigBee/BT-LE was used in Securitas Direct AB (Verisure) sensor network. However, changing into using only ZigBee/BT-LE would be impossible in terms of development cost and operabilitiy with legacy products. Since the gateway of the system handles all the communication, even with old nodes, changing the protocol of the sensor network would be costly and it would result in that one can not communicate with old nodes.

By adding an extra module into the gateway that handles the ZigBee/BT-LE nodes and keep the current Securitas Direct Safety Radio Protocol for the security mechanism, would be the best way to implement this feature. This could also be the best way to be interoperable with other health devices and also be the most cost effective and reliable way to get a Continua Health certification.

5.4.3 BT-LE or ZigBee

ZigBee and Bluetooth are similar in many ways, however one great feature of ZigBee is the mesh capabilities which makes it possible for a ZigBee network to become very large. It also has lower power consumption and have a greater range, but with a lower bandwidth. ZigBee is also more known and has a number of

application profiles, in our case the most promising is the Health Care profile that makes it ISO/IEEE 11073 PHD compliant.

In terms of choosing between Bluetooth-LE and ZigBee, there are no real advantages in any of them if one only take into consideration that it should be compatible with ISO/IEEE 11073 PHD. However, in terms of future capabilities, going with the ZigBee alliance is proposed. Mainly because BT-LE works as a Personal Area Network and ZigBee is considered a Sensor Network.

Table 5.2: ZigBee and Bluetooth LE comparison

Type	**ZigBee (802.15.4)**	**Bluetooth LE (802.15.1)**
Application Focus	Monitoring and Control	Cable Replacement
System resources	32KB-64KB+	250KB+
Battery Life	100-1000+ days	1-7 Days
Network Size	Unlimited	7
Bandwidth	20-250 KB/s	720 KB/s
Transmission Range	1-100+ meters	1-10+ meters
Success metrics	Reliability, Power, Cost	Cost, Convenience

5.4.4 The Remote Service

The remote service plays significant part in the system, see Figure 5.9. To be able to send the data in a secure way, one needs to focus on the different computer security aspects, such as confidentiality on the data, integrity on the transmitted data and non-repudiation on the data transactions in order to make a secure way for sensitive data to pass through systems. To be able to guarantee that the data is sent to and from the health device in the home in a secure manner is crucial. The remote central could for example have services that provides surveillance on specific devices, for example a pulse meter. This would however be a quite costly service when you take the cost for the personal into the calculation.

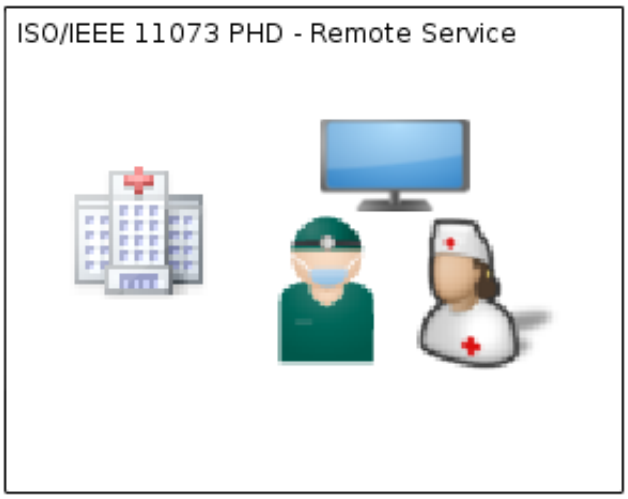

Figure 5.9: The Remote Service, not defined by ISO/IEEE 11073 PHD

Chapter 6

Conclusion

In this thesis we have investigated into the possibilities to send medical data through the existing sensor network. This was shown successful quite early but we had to make the sensor network compatible with medical standards and to make the system future-proof. We looked into other modern sensor network standards that could prove to be feasible in future collaborations, not only with a medical profile.

The implementation we did was based on sending ASCII commands to the device. We were able to send data to the device by doing API-calls through a REST-client we made. It sends data by disguising ASCII decimal values, in packets of three bytes, where other device configuration parameters should be.

We implemented the up link by utilizing an existing command in the sensor network which was used for testing and sending error commands. This sent the data, in packets of four bytes, to the database where we could query the data sent and extract the information we wanted.

The major restriction of the development we encountered was the lack of application layer i.e. there is no way in the sensor network protocol to create a raw data link, instead all the data sent is created with known commands. In order to create a new command one needs to do an update of the gateway and create new API calls on the back end. These kind of implementations are not possible because it involves a large number of people needing to cooperate.

A major problem is the transmission from the application on the other side of the back end. Since we can only send three bytes at each REST call, and a REST call takes $\sim 1s$, this makes 100 bytes of data transmission taking almost one minute with the loss of packets in the UDP connection and delays in the sensor networks. This was shown to be ineffective if we try to put it into a larger perspective.

There are streaming capabilities of the sensor network that we worked with,

however, in the beginning of the project there was no way for us to implement or use such transmission due to t. This would have allowed us to send the data in a more reliable way rather than with the one-sided retransmission algorithm that we use at the moment.

As previously mentioned numerous attempts have been made to accomplish medical specific transmission by using wireless protocols and back end solutions. However this task has not been easy, as for the number of different approaches are equivalent to the number of different solutions to this type of application. However, a common search for a standard has arised, thus indicating the need for a common goal. The problem derives from short term thinking when developing where the medical technical device serves its purpose for the moment, but in reality lacks support or compatibility for future products of same type. The surge for a shared goal has been realised the past years under the name ISO/IEEE 11073 PHD to accommodate the requirements given. It is however up to the manufacturer or developer to decide whether they wish to comply with these standards in order to reach levels of certification of these devices.

As for today, certification level for medical equipment can be reached. One of the research objectives by the ZigBee® alliance has been to look into the possibilities that involves standards promoting and supporting data exchange between medical and non medical devices. This has resulted in the so called ZigBee® Health care public application profile which has been developed with the purpose to support medical attention in non-invasive contexts.

The public application profile refers to the outline of the application specific requirements in which the planned future medical devices will communicate upon. They consist of agreements regarding the messages types, message formats and proceeding actions. These support the developer to create application entities that inter-operate on different devices, thus allowing for easier interoperability. The application specific commands that are handled by the device manufacturer involve sending, request data and process data commands that are communicated over the ZigBee stack architecture.

The project has not focused on how Securitas Direct AB (Verisure) should forward the collected data to other, remote services such as hospital or some other health center. This project has only looked into how the sensor network could handle such data. Future work would focus on how the data should be delivered from the database of Securitas Direct AB (Verisure). This would probably be done with a API call that allows for extraction of the data.

One research area that have not been investigated is the scalability of implemented system. There are configuration possibilities in the code where you can set the serial number of the device responsible for the medical equipment, but no

research what so ever have bean laid on this matter.

Bibliography

[1] ZigBee Alliance. "ZigBee document specification". In: *ZigBee Document 053474r17*. January 17, 2008.

[2] M. Benner and L. Schope. "Using Continua Health Alliance Standards - Implementation and Experiences of IEEE 11073". In: *Mobile Data Management (MDM), 2011 12th IEEE International Conference on*. Vol. 2. 2011, pp. 40–45. DOI: 10.1109/MDM.2011.25.

[3] M. Clarke et al. "Building point of care health technologies on the IEEE 11073 health device standards". In: *Point-of-Care Healthcare Technologies (PHT), 2013 IEEE*. 2013, pp. 117–119. DOI: 10.1109/PHT.2013.6461298.

[4] Anders Ekholm. "Den ljusnande framtid är vård, delresultat från LEV-projektet". In: (2010).

[5] Fujuan Guo et al. "System design for human vital signals monitor based on wireless sensor network". In: *Measurement, Information and Control (MIC), 2012 International Conference on*. Vol. 1. 2012, pp. 360–364. DOI: 10.1109/MIC.2012.6273271.

[6] "Health Informatics-Personal Health Device Communication Part 20601: Application Profile- Optimized Exchange Protocol". In: *IEEE Std 11073-20601-2008* (2008), pp. c1–198. DOI: 10.1109/IEEESTD.2008.4723887.

[7] Hemocue. "Document specification FSF0210/1". In: *Description of the HemoCue 201+ Communication Protocol*. January 29, 2010.

[8] European Telecommunications Standards Institute. “ETSI EN 300 220-1 Version 2.4.1”. In: (2012-01).

[9] Texas Instruments. “SimpliciTI: Simple Modular RF Network Developers Notes 1.40”. In: *SimpliciTI: Simple Modular RF Network Developers Notes Specification*. 2009.

[10] Texas Instruments. “SimpliciTI: Simple Modular RF Network Specification 1.09”. In: *SimpliciTI: Simple Modular RF Network Specification*. 2007-2009.

[11] Boris Magnusson. *PalCom - middleware for Pervasive Computing @ONLINE*. URL: http://palcom.cs.lth.se/Palcom/Palcom.html.

[12] Jae-Choong Nam et al. “Design and development of a u-Health system based on the ISO/IEEE 11073 PHD standards”. In: *Communications (APCC), 2011 17th Asia-Pacific Conference on*. 2011, pp. 789–793. DOI: 10.1109/APCC.2011.6152915.

[13] Se-Jin Oh and Chae-Woo Lee. “u-Healthcare SensorGrid Gateway for connecting Wireless Sensor Network and Grid Network”. In: *Advanced Communication Technology, 2008. ICACT 2008. 10th International Conference on*. Vol. 1. Feb. Pp. 827–831. DOI: 10.1109/ICACT.2008.4493882.

[14] B. Piniewski et al. “Empowering Healthcare Patients with Smart Technology”. In: *Computer* 43.7 (2010), pp. 27–34. ISSN: 0018-9162. DOI: 10.1109/MC.2010.200.

[15] L. Skrzypczak, D. Grimaldi, and R. Rak. “Analysis of the different wireless transmission technologies in Distributed Measurement Systems”. In: *Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2009. IDAACS 2009. IEEE International Workshop on*. 2009, pp. 673–678. DOI: 10.1109/IDAACS.2009.5342890.

[16] M. Spadacini et al. “Wireless networks for smart surveillance: Technologies, protocol design and experiments”. In: *Wireless Communications and Networking Conference Workshops (WCNCW), 2012 IEEE*. 2012, pp. 214–219. DOI: 10.1109/WCNCW.2012.6215493.

[17] Inc. Texas Instruments. "SimpliciTI: Simple Modular RF Network Developers Notes 1.40 Revised to reflect SimpliciTI 1.1.0." In: *SimpliciTI: Simple Modular RF Network Developers Notes*. 3/24/2009.

[18] F. Wartena, J. Muskens, and L. Schmitt. "Continua: The Impact of a Personal Telehealth Ecosystem". In: *eHealth, Telemedicine, and Social Medicine, 2009. eTELEMED '09. International Conference on.* 2009, pp. 13–18. DOI: 10.1109/eTELEMED.2009.8.

[19] F. Wartena et al. "Continua: The reference architecture of a personal telehealth ecosystem". In: *e-Health Networking Applications and Services (Healthcom), 2010 12th IEEE International Conference on.* 2010, pp. 1–6. DOI: 10.1109/HEALTH.2010.5556588.